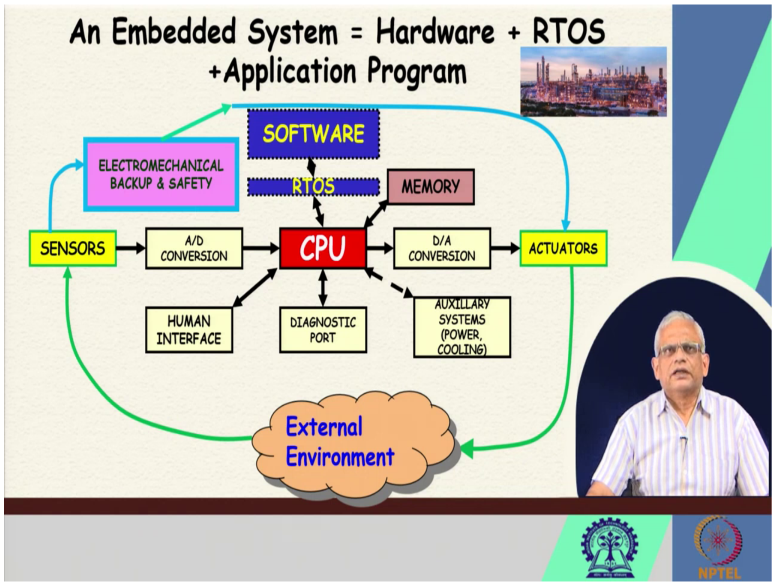
Real Time Systems
Professor Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture 03
Characteristics of a real-time embedded system

Welcome to this course on Real Time Systems. In the last lecture, we had a very brief introduction to what is real time, real time systems and so on. And today is also an introductory lecture. Continuation from the last lecture, we will see the characteristics of real time systems in more detail in today's lecture. Let us get started.

(Refer Slide Time: 00:43)



An Embedded System = Hardware + RTOS +Application Program

We had said last time that real time system is typically an embedded system, even though not always, but typically an embedded system and many times it controls its environment. This is a chemical plant, large chemical plant, but controlled by an embedded system. The embedded system if we look at it, it has hardware, a real time operating system and an application program which is a real time program. In this course, our focus is on the real time operating system and the real time application program. And together this hardware real time operating system and application program control this large chemical plant.

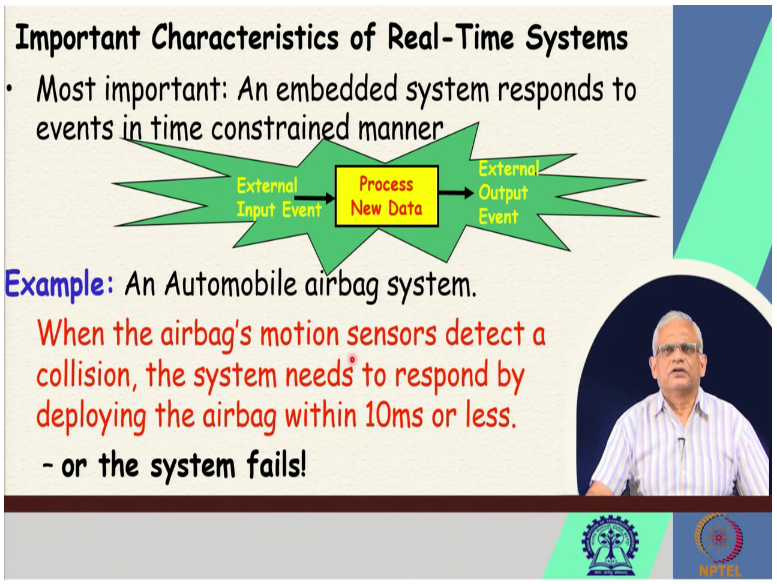
If we blow up the embedded system that is sitting inside this chemical plant, we will see that many our hardware components, the ones that are marked yellow and red here are the hardware parts, the sensors, which collect various information from the chemical plant and the sensor input is given to analog to digital converter and which is interface to the CPU. And similarly, the CPU output through a digital to analog conversion is given to an actuator. The actuator here maybe a valve control or a suites and we have a human interface, where the operator can query various parameters, adjust the settings of the chemical plant and so on.

We have a diagnostic port through which diagnostic equipment can be attached to the embedded system to collect diagnostic information. And of course, there is auxiliary hardware like power supply, cooling and so on. And the memory is interfaced to the CPU. That is also a hardware component. And then we have the real time operating system running on the CPU, and the application software which is real time software is using the services and have real time operating system and it is running, we can say that it is running on top of the real time operating system.

The actuator carries out various actions on the environment which is the chemical plant or maybe any other system and the parameters of this plant are read by the sensors. And see here that there is a electromechanical backup and safety. In case there is a failure in the hardware or software, the electromechanical backup and safety gets activated and provides a failsafe operation of the chemical plant. That is even when the embedded system fails the chemical plant does not fail. Maybe it just undergoes a graceful shutting down the electromechanical backup and safety may do that.

So, even if the system fails, there is an arrangement here mechanical backup and safety which provides the necessary safety and backup. So, this is the overall picture that captures the functioning of this real time operating system, real time application, which is the main focus of our course here. But just to be aware in the context in which they work, we need to understand this diagram.

(Refer Slide Time: 05:34)



Important Characteristics of Real-Time Systems

- Most important: An embedded system responds to events in time constrained manner

External Input Event
Process New Data
External Output Event

Example: An Automobile airbag system.

When the airbag's motion sensors detect a collision, the system needs to respond by deploying the airbag within 10ms or less.

- or the system fails!

Now, let us look at some of the important characteristics of these embedded systems. The embedded system gets data from various sensors and these are the input events and it produces output events which are basically commands to the actuators. And the green one that you see here, that is the environment; it is sitting inside its environment and controlling the environment.

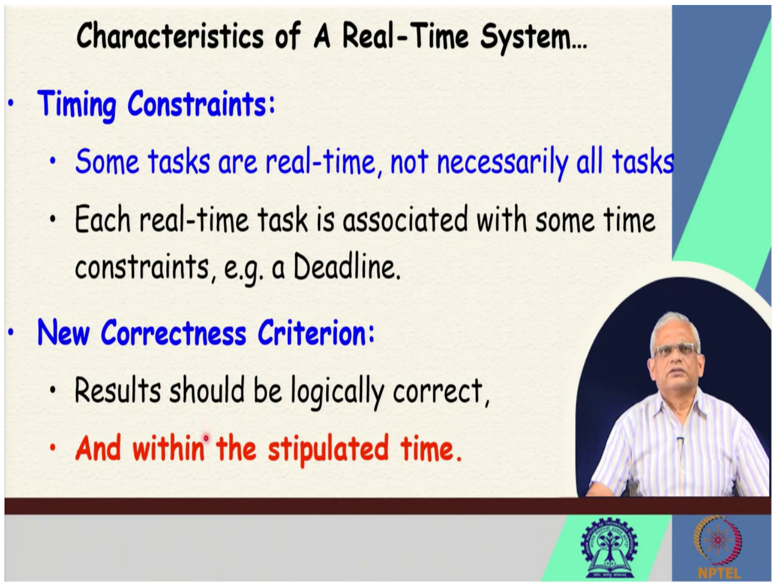
So, we can say that the most important characteristic of an embedded real time system is that it responds to events in the environment, the various events that arise in the environment in a time constrained manner. The events may be that the temperature exceeds some value; the pressure becomes low etc., etc. So, once those events occur these are handled by the processor, takes corrective action through the actuators. But then these are done in a time constrained manner.

One example of such a system is an automobile airbag system. Whenever there is an accident involving an automobile, the airbag has to inflate almost instantly, maybe let us say within 30 milliseconds. So, the accident is detected by the motion sensor. When there is a sudden deceleration, unnatural deceleration which indicates an accident, in that case the system deploys the airbag within let us says 10 milliseconds or less time.

So, the system here detects when to react, that is when the deceleration of the vehicle is so large that it can be classified as a, or it can be considered as an accident. And that is done very fast. It responds within 10 milliseconds. If it inflects any later, it may not serve its purpose. We will say that the airbag system would have failed. So, here the time constraint within which the action is to occur is 10 milliseconds within which the airbag has to be inflated. That is the event of deceleration, sensing the deceleration, deciding to deploy the air put all this has to be done within 10 milliseconds.

If it does not inflate in 10 milliseconds, we will say that, even if it let us say deploys after 50 milliseconds, at 50 milliseconds it deploys, even though it correctly deployed the airbag, but it was too late. We will say that the system failed to deploy the airbag in time. So, that is the time constraint here which has to be obeyed. And the correct working of the system is not only about just deploying the airbag, but deploying within the specified time. That is an important characteristic of real time systems.

(Refer Slide Time: 09:44)



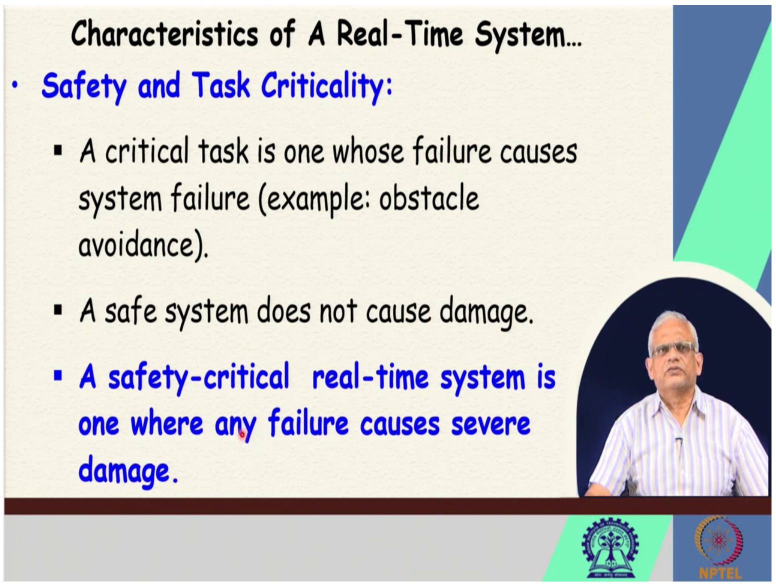
Characteristics of A Real-Time System…

- Timing Constraints:
  - Some tasks are real-time, not necessarily all tasks
  - Each real-time task is associated with some time constraints, e.g. a Deadline.
- New Correctness Criterion:
  - Results should be logically correct,
  - And within the stipulated time.

Now, let us look at other characteristic. Another is time constraint. We just saw that many of the actions have to be done in a time constraint way. But of course, not all tasks in a real time system will have time constraints. There may be tasks which may be doing logging, answering or responding to user queries and so on which would not have time constraints. So, the important thing to note here is that different tasks might have different time constraints, some tasks may not have time constraints. And this time constraint is typically in the form of a deadline by which the action needs to be completed.

This leads us to a new correctness criterion for these systems. We say that these systems are working correctly, if not only their results are functionally correct, but also produced within the required time. So, the results are logically correct and also are carried out within the stipulated time. This is in contrast to a traditional system, where as long as the results produced are logically correct we say that the system is working correctly. But here not only that it has to be logically correct, but also completed within the stipulated time.

(Refer Slide Time: 11:28)



Characteristics of A Real-Time System…

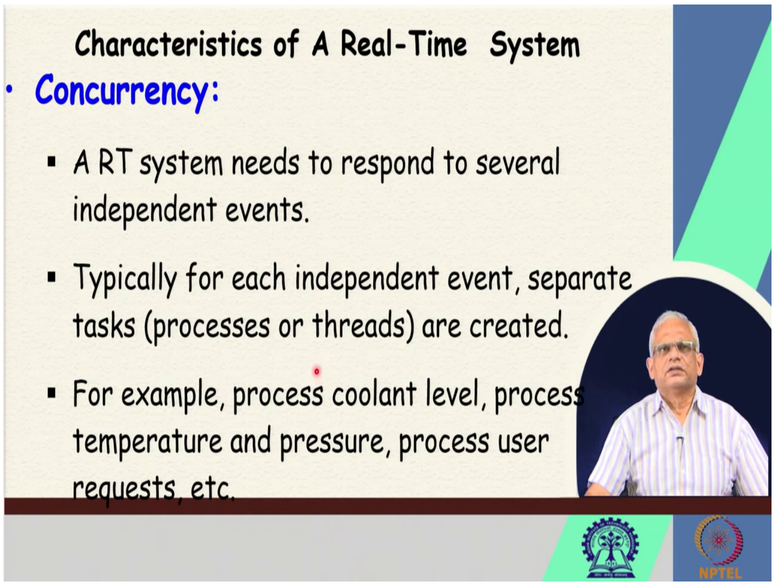
- Safety and Task Criticality:
  - A critical task is one whose failure causes system failure (example: obstacle avoidance).
  - A safe system does not cause damage.
  - A safety-critical real-time system is one where any failure causes severe damage.

Another important characteristic of these real time systems is safety and task criticality. Some of the tasks in real time systems would be critical tasks. For these tasks, if there is some failure, for example, there is a logically incorrect result or maybe the result is too late, then the system fails. For example, a robot is moving on the terrain and there is an obstacle and one of the tasks of the robot is obstacle avoidance.

Now, if the obstacle avoidance task produces result for some reason very late, then it serves no purpose. Even though it was a correct result which directed the robot to move away from the obstacle, but it is of little use. The robot would have already collided with the obstacle by them by that time. So, these tasks we say that these are critical tasks. Any failure here will cause the failure of the system.

Now, let us look at the other concept of a safe system. A safe system even if fails does not cause any damage. But many of these embedded real time systems are safety critical in nature. This is a term which will define the safety critical nature of a real time system means that any failure causes severe damages. The system fails and any system failure causes severe damage. So, these are not safe systems in the sense that a failure will cause damage. The critical tasks have to be completed and within time, otherwise there will be damage. These are not really safe systems, especially with respect to the critical tasks and we say that these are safety critical systems.

(Refer Slide Time: 14:18)



Characteristics of A Real-Time System

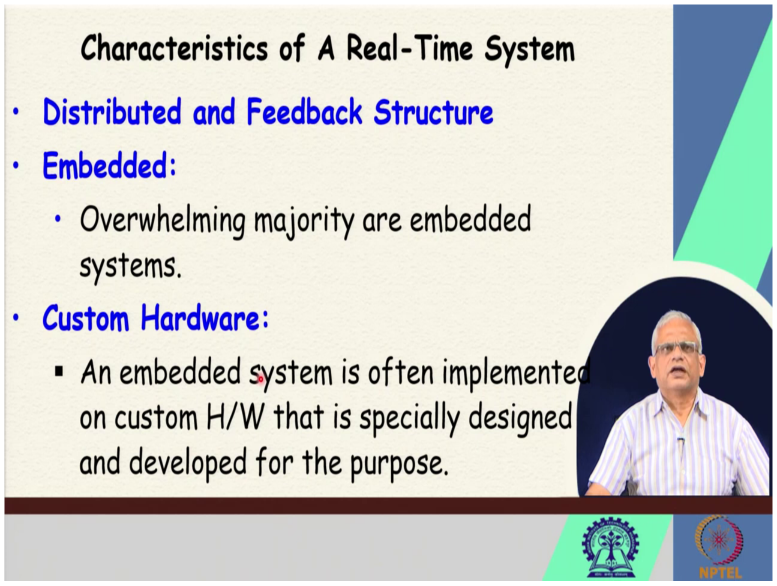
- Concurrency:
  - A RT system needs to respond to several independent events.
  - Typically for each independent event, separate tasks (processes or threads) are created.
  - For example, process coolant level, process temperature and pressure, process user requests, etc.

Another important aspect of the real time system says almost always these are concurrent. These are concurrent a typical real time system. Of course, extremely simple, trivial systems exist, which respond to only one event or a set of related events which may not be concurrent. As we proceed in this course, we will look at that. But most non-trivial real time embedded systems are concurrent. And the main reason why these needs to be concurrent is that there are several events to which it the system needs to respond.

And typically, whenever an event occurs, it creates a task or a process or a thread to handle that event. We say that a process instance or a task occurs to handle a specific event. For every event, there is a task that is initiated, which is an instance of a process or a thread. Just to consider an example, let us say we have a coolant level indicator and the coolant level let us say goes low and there is an event. So, to handle the coolant level low event a task will be initiated which is basically a process or a thread.

And similarly, another event may be concerning temperature and pressure and if there is a variation, there will be a task initiated to handle this. Similarly, another processor thread may be initiated to handle the user requests. So, almost always these real time systems are concurrent. We will have many instances of tasks executing concurrently or many instances of threads which are basically initiated for these tasks and these proceed concurrently. So, this is an important characteristic of real time systems.

(Refer Slide Time: 16:59)



Characteristics of A Real-Time System

- **Distributed and Feedback Structure**
- **Embedded:**
  - Overwhelming majority are embedded systems.
- **Custom Hardware:**
  - An embedded system is often implemented on custom H/W that is specially designed and developed for the purpose.

Another important characteristic is the distributed and feedback structure. Just think of the chemical plant we are discussing. There are events occurring at different places which are quite apart geographically maybe several 100 meters apart or maybe several kilometers apart. Now, the real time system cannot be just one centralized small system to be able to handle such extended environment. We need to have distributed processing.

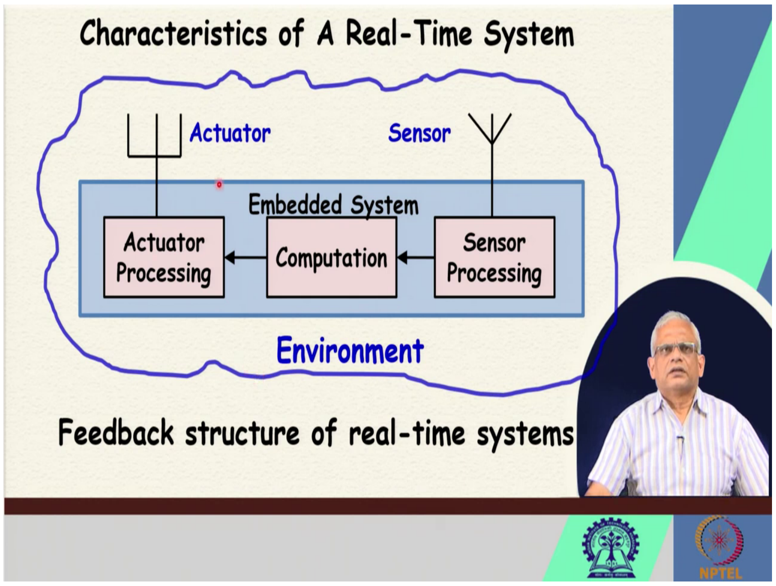
The real time system must have parts or components that are collocated with places where these events occur and take some action locally and also communicate with the other components. And these are feedback in nature, because as the event occurs some actions are taken place. And the effect of the action changes the environment and which is again sensed and so on. So, there is a feedback structure.

And many of these real time systems as we have been saying are embedded in nature. And often, these systems do not run on the traditional hardware like let us say a laptop or a desktop or a server, these run on special embedded hardware, maybe a board with a processor, memory and so on. And many times the hardware is developed specifically for a system and therefore we call it as custom hardware.

To develop such systems, we need to develop both the hardware and software. Because the hardware is to be developed specifically for this system cannot just buy off the shelf, the hardware is developed and also software is developed. And to minimize the development time, often the hardware and software are developed together concurrently. So, the software is developed on some system, let us say, a desktop or a laptop. And as the hardware gets ready, this is run tested on the hardware by that time software development is complete.

In this course, we are not really going to discuss too much on development of the concurrent hardware and software. But since we are discussing about real time embedded systems, we just have a passing remark or just a rough understanding of how these custom hardware and software are developed.

(Refer Slide Time: 20:27)



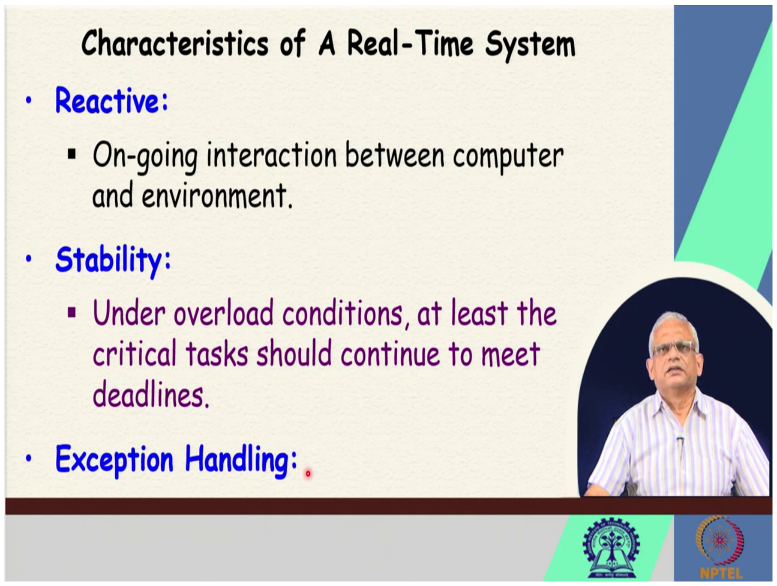
Characteristics of A Real-Time System

Feedback structure of real-time systems

Now, these systems as we have been saying have a feedback in nature. So, here the characteristics of the environment are sensed through a sensor and then these are passed to the processor which produces some results and the actuator processes the commands given by the embedded system. And these actuators take some action maybe open some valves, close some valves or switch on the heater, switch off etc.

Now, that impact the environment and the events they change their characteristics, and these new events are again sensed. So, there is a feedback structure here from sensing the event to taking the action and the action having effect on the environment and new events getting detected and so on. So, many of these systems we are talking of are, have a inherent feedback structure.

(Refer Slide Time: 21:47)



Characteristics of A Real-Time System

- **Reactive:**
  - On-going interaction between computer and environment.
- **Stability:**
  - Under overload conditions, at least the critical tasks should continue to meet deadlines.
- **Exception Handling:**

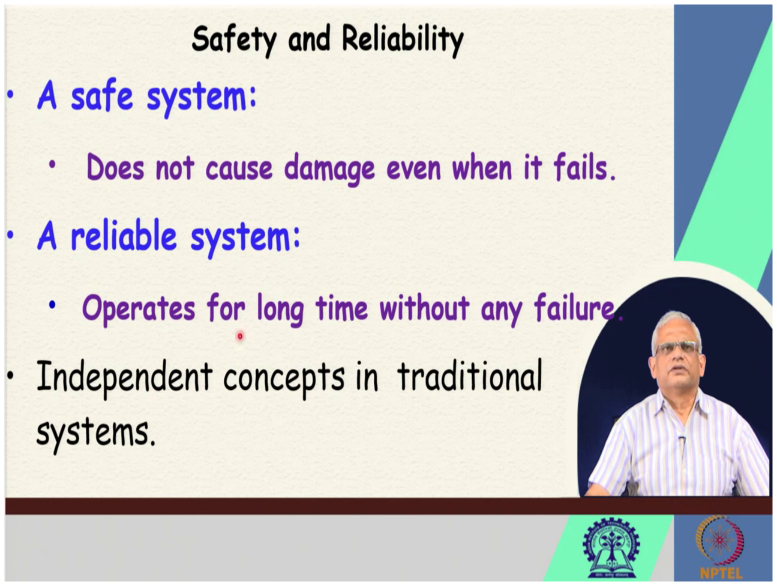
Another important characteristic of this system which set this apart from the traditional systems is the reactive in nature. I think in the last class, we are discussing about a reactive system where there is not a simple function computation like in traditional software, where we give input and produce output, but here it is reactive and there is an ongoing reaction between the computer or the real time system and the environment. There is an ongoing reaction or interaction between the computer and the environment.

Another important characteristic which we will often refer to in this course as we proceed is about stability. Now, let us try to define this term stability, which means that even if there is an overload situation like many events, they occurred together a fire alarm event and a temperature exceeding event and pressure exceeding event and so on occurred in a chemical plant, there is an overload in the embedded system.

But even when such overload situations occur, that too many events have occurred at the same time and many tasks have to be created to handle them. But in this situation also at least the critical tasks should continue to meet their deadlines. Maybe the tasks which are not critical like let us say response to user queries or maybe logging activities etc. may get delayed a little bit, but then the critical tasks should continue to meet their deadline. So, that is defined as the stability.

Let me just repeat that again. Stability means even when the system gets overloaded for some reason, the critical tasks should continue to meet their deadlines. Another characteristic is exception handling. Whenever there are exception conditions which occur like there is a power failure and so on, these have to be handled satisfactorily. So, the exception handling mechanism must be incorporated in the system.

(Refer Slide Time: 24:26)



Safety and Reliability

- A safe system:
  - Does not cause damage even when it fails.
- A reliable system:
  - Operates for long time without any failure.
- Independent concepts in traditional systems.

Now, let us just try to explore these terms safety and reliability, because these are crucial to understanding some characteristic of the real time systems. The real time systems are safe systems. They have to be safe systems. And we said that a safe system is one which does not cause damage even when it fails. Just think of a chemical plant or a nuclear reactor.

Now, even due to some reason, let us say the software failed or the hardware failed, but then the plant should not cause damage. It should have a graceful shutdown, exit or whatever. It should not cause damage and that we say is a failsafe, is a safe system.

Another notion here is a reliable system. A reliable system operates for long time without any failure. So, that is the definition of a reliable system. A reliable system can operate for long times, maybe millions of hours without encountering a single failure, and then we say that the system is reliable.

Of course, a reliable system may appear to be a safe system, but not necessarily. A reliable system can also be unsafe. These two are independent concepts. A reliable system can be unsafe, and a safe system can be unreliable.

Now, let us explore this further about a safe system and a reliable system, because these are crucial concepts, the safety and reliability of a real time embedded system. We are almost at the end of this lecture. We will stop here and we will continue from this point and we will explore further about safety and reliability of a real time system before proceeding to further details about these systems. Thank you.